Reproducible Research - Peer Assessment Project 1

TLK

October 2, 2017

### Read csv file

activity <- read.csv("activity.csv")  
head(activity)

## steps date interval  
## 1 NA 2012-10-01 0  
## 2 NA 2012-10-01 5  
## 3 NA 2012-10-01 10  
## 4 NA 2012-10-01 15  
## 5 NA 2012-10-01 20  
## 6 NA 2012-10-01 25

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.4.1

library(plyr)

## Warning: package 'plyr' was built under R version 3.4.1

Processing the Data

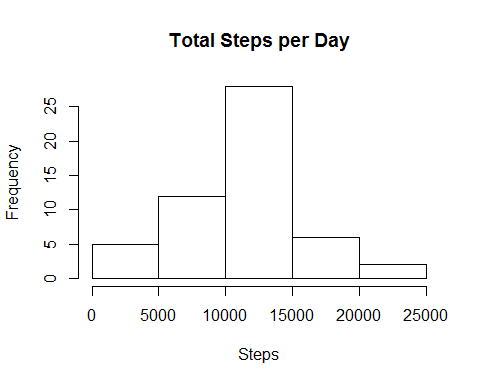
activity$day <- weekdays(as.Date(activity$date))  
activity$DateTime<- as.POSIXct(activity$date, format="%Y-%m-%d")  
  
##pulling data without nas  
clean <- activity[!is.na(activity$steps),]

Calculate the total number of steps taken per day

## summarizing total steps per date  
sumTable <- aggregate(activity$steps ~ activity$date, FUN=sum, )  
colnames(sumTable)<- c("Date", "Steps")

Make a histogram of the total number of steps taken each day

## Creating the histogram of total steps per day  
hist(sumTable$Steps, breaks=5, xlab="Steps", main = "Total Steps per Day")



dev.copy(png,'histogram1.png')

## png   
## 3

dev.off()

## png   
## 2

Calculate and report the mean and median of the total number of steps taken per day

## Mean of Steps  
as.integer(mean(sumTable$Steps))

## [1] 10766

## Median of Steps  
as.integer(median(sumTable$Steps))

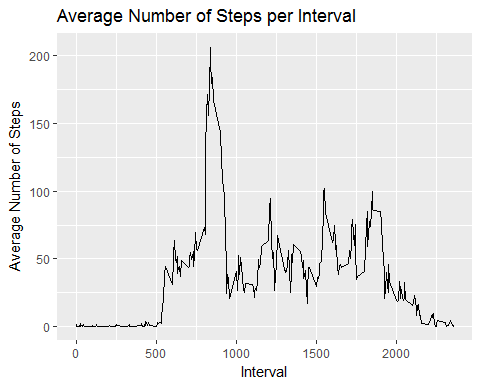
## [1] 10765

The average number of steps taken each day was 10766 steps.

The median number of steps taken each day was 10765 steps

Make a time series plot of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all days (y-axis)

library(plyr)  
library(ggplot2)  
##pulling data without nas  
clean <- activity[!is.na(activity$steps),]  
  
##create average number of steps per interval  
intervalTable <- ddply(clean, .(interval), summarize, Avg = mean(steps))  
  
##Create line plot of average number of steps per interval  
p <- ggplot(intervalTable, aes(x=interval, y=Avg), xlab = "Interval", ylab="Average Number of Steps")  
p + geom\_line()+xlab("Interval")+ylab("Average Number of Steps")+ggtitle("Average Number of Steps per Interval")



dev.copy(png,'lineplot1.png')

## png   
## 3

dev.off()

## png   
## 2

Which 5-minute interval, on average across all the days in the dataset, contains the maximum number of steps?

##Maximum steps by interval  
maxSteps <- max(intervalTable$Avg)  
##Which interval contains the maximum average number of steps  
intervalTable[intervalTable$Avg==maxSteps,1]

## [1] 835

The maximum number of steps for a 5-minute interval was 206 steps.

The 5-minute interval which had the maximum number of steps was the 835 interval.

Calculate and report the total number of missing values in the dataset (i.e. the total number of rows with NAs)

##Number of NAs in original data set  
nrow(activity[is.na(activity$steps),])

## [1] 2304

My strategy for filling in NAs will be to substitute the missing steps with the average 5-minute interval based on the day of the week.

## Create the average number of steps per weekday and interval  
avgTable <- ddply(clean, .(interval, day), summarize, Avg = mean(steps))  
  
## Create dataset with all NAs for substitution  
nadata<- activity[is.na(activity$steps),]  
## Merge NA data with average weekday interval for substitution  
newdata<-merge(nadata, avgTable, by=c("interval", "day"))

Create a new dataset that is equal to the original dataset but with the missing data filled in.

## Reorder the new substituded data in the same format as clean data set  
newdata2<- newdata[,c(6,4,1,2,5)]  
colnames(newdata2)<- c("steps", "date", "interval", "day", "DateTime")  
  
##Merge the NA averages and non NA data together  
mergeData <- rbind(clean, newdata2)

Make a histogram of the total number of steps taken each day and Calculate and report the mean and median total number of steps taken per day.

##Create sum of steps per date to compare with step 1  
sumTable2 <- aggregate(mergeData$steps ~ mergeData$date, FUN=sum, )  
colnames(sumTable2)<- c("Date", "Steps")  
dev.copy(png,'histogram2.png')

## png   
## 3

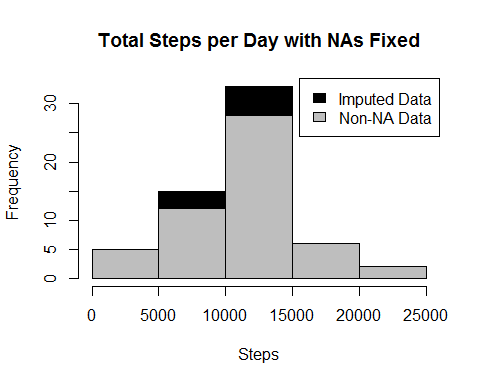
dev.off()

## png   
## 2

## Median of Steps with NA data taken care of  
as.integer(median(sumTable2$Steps))

## [1] 11015

## Creating the histogram of total steps per day, categorized by data set to show impact  
hist(sumTable2$Steps, breaks=5, xlab="Steps", main = "Total Steps per Day with NAs Fixed", col="Black")  
hist(sumTable$Steps, breaks=5, xlab="Steps", main = "Total Steps per Day with NAs Fixed", col="Grey", add=T)  
legend("topright", c("Imputed Data", "Non-NA Data"), fill=c("black", "grey") )



dev.copy(png,'histogram3.png')

## png   
## 3

dev.off()

## png   
## 2

The new mean of the imputed data is 10821 steps compared to the old mean of 10766 steps. That creates a difference of 55 steps on average per day.

The new median of the imputed data is 11015 steps compared to the old median of 10765 steps. That creates a difference of 250 steps for the median.

However, the overall shape of the distribution has not changed.

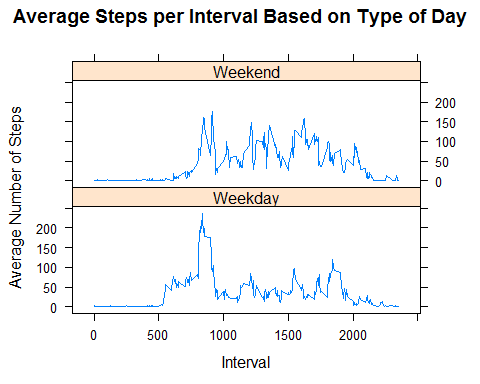
Create a new factor variable in the dataset with two levels - ???weekday??? and ???weekend??? indicating whether a given date is a weekday or weekend day.

## Create new category based on the days of the week  
mergeData$DayCategory <- ifelse(mergeData$day %in% c("Saturday", "Sunday"), "Weekend", "Weekday")

Make a panel plot containing a time series plot (i.e. type = ???l???) of the 5-minute interval (x-axis) and the average number of steps taken, averaged across all weekday days or weekend days (y-axis).

library(lattice)

## Summarize data by interval and type of day  
intervalTable2 <- ddply(mergeData, .(interval, DayCategory), summarize, Avg = mean(steps))  
  
##Plot data in a panel plot  
xyplot(Avg~interval|DayCategory, data=intervalTable2, type="l", layout = c(1,2),  
 main="Average Steps per Interval Based on Type of Day",   
 ylab="Average Number of Steps", xlab="Interval")



dev.copy(png,'panelplot1.png')

## png   
## 3

dev.off()

## png   
## 2

Yes, the step activity trends are different based on whether the day occurs on a weekend or not.